Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 14 (2012) 74 – 82

Procedia Computer Science

Proceedings of the 4th International Conference on Software Development for Enhancing Accessibility and Fighting Info-exclusion (DSAI 2012)

Realtime local navigation for the blind: detection of lateral doors and sound interface

M. Moreno, S. Shahrabadi, J.José, J.M.H. du Buf , J.M.F. Rodrigues

Vision Laboratory, LARSyS, University of the Algarve, Campus de Gambelas, 8005-139 Faro, Portugal

Abstract

Worldwide there are about 285 million visually impaired persons, of which 39 million are blind and the others have low vision. Almost all systems designed to assist them are quite complex and expensive, but most blind persons do not have advanced technical assistance and they are rather poor. We are therefore developing a low-cost navigation aid which can be afforded by almost all blind persons: basically, the ultimate goal is to use only a mobile phone with a built-in camera. This aid complements the white cane, it is easily portable, and it is not a hindrance when walking with the cane. The system will have an easy and intuitive interface, yet providing assistance in local and global navigation in realtime. In this paper we present the progress concerning local navigation. Path and obstacle detection just beyond the reach of the cane is now supplemented by detection of doors in corridors. This is necessary for localization, i.e., for developing a better impression of the environment and for finding a specific room. A sophisticated sound interface can assist the user for centering on paths like sidewalks and corridors, alerting to looming obstacles for avoiding them.

Keywords: Vision, Blind, Navigation, Doors, Audio interface.

1. Introduction

Worldwide there are about 285 million visually impaired persons. About 39 million are completely blind and 246 million have low vision; see [1] for more details. Most persons must rely on the white cane during local navigation, constantly swaying it in front for negotiating walking paths and obstacles in the immediate vicinity.

Several approaches try to ease a blind person's everyday life by using assistive technologies. One named "Smart-cane" [2] is an electronic cane with built-in ultrasound sensors for detecting obstacles, both horizontally and vertically. Its cost, estimated at USD 35, is affordable even by persons in poor countries. Schmitz et al. [3] developed a navigation system that seamlessly integrates static maps with dynamic, location-based textual information from a variety of sources. Each information source requires a different acquisition technique. All acquired information is combined by a context management platform, and then presented to the user as a tactile or acoustic map depending on the sources available at the current position and time. Positioning is achieved by

 doi:10.1016/j.procs.2012.10.009

a combination of an inertial tracking system, RFID technology and GPS, and the user is guided to a desired destination by speech output and a haptic cane. This system is, in contrast to the "Smartcane," much more complex and less affordable. In [4] an algorithm is presented to recognize landmarks suitably placed on sidewalks. This algorithm uses a combination of Peano-Hilbert space-filling curves for dimension reduction of image data, and ensemble empirical mode decomposition (EEMD) to pre-process images, which yields a fast and efficient recognition method. For previous work on the detection of doors we refer to [5].

The system presented here is part of a larger project entitled "Blavigator: a cheap and reliable navigation aid for the blind." Financed by the Portuguese Foundation for Science and Technology, it combines several technologies, such as GPS, GIS, WiFi and computer vision, to create a system which assists visually impaired persons in navigating both in- and outdoor. Its goal is to develop a vision and navigation aid which is: (a) not expensive, because about 90% of potential users live in so-called developing countries; (b) easily portable, not being a hindrance when walking with the cane; (c) complementing the cane, but not substituting it because blind persons must always be able to rely on the cane; (d) extremely easy to use in terms of intuitive interfacing; (e) simple to assemble, install and operate, without need for very skilled technicians; and (f) providing useful assistance for local and global navigation in realtime. Blavigator, from blind navigator, is a follow-up project of SmartVision. For further information about the SmartVision prototype we refer to [6].

Below, we briefly present the methods developed for SmartVision: the vision system which complements the white cane, with obstacle detection beyond the reach of the cane, and providing help for centering on walkable paths. We then focus on the new contributions: detection of doors in corridors, in the future also of houses along sidewalks, and a newly developed sound interface for path centering, obstacle alerts and obstacle avoidance.

2. Path and Obstacle Detection

In a previous paper we presented path and obstacle detection [7]. The system is based on a camera worn at the chest of the user, at a height of about 1.5 m from the ground. The height of the camera depends on the height of the user, but it is not relevant to the system's performance. Even a mobile phone with built-in camera can be used, as nowadays most models have a VGA or better resolution. The resolution must be sufficient to resolve textures of pavements and potential obstacles like holes with a minimum size of 10 cm at a distance of 2 to 5 meters from the camera.

The camera points forward, the image plane being vertical, but deviations due to swaying during walking are not problematic because problems are solved in the image processing steps.

The goal is to guide the user on walkable paths, i.e., the area where the user can walk safely. Examples are indoor corridors and outdoor sidewalks, but any path with a left and a right border can be detected. Because of perspective projection, the left and right border and other parallel lines intersect at a point called the vanishing point (VP). This point is first used to select the part of the image which is used for detecting valid borders, i.e., below the VP. After pre-processing using the Canny edge detector [8], an adapted version of the Hough transform [9] is applied to extract the left and right borders from the image. Since vertical camera alignment is not fixed but varies over time when the user walks, we use the VP in order to determine the horizon line (HL). The height of HL is computed dynamically, by averaging the values of the previous five frames. The area inside the left and right borders and the bottom line of the image we call the path window (PW). For more details concerning path detection we refer to [7].

The PW is wider than the area in front where the blind person will walk and where obstacles must be detected. Hence, the PW is narrowed by drawing new lines through the VP: the positions of the original borders of the PW at the bottom line are shifted right (left border) and left (right border), which results in a narrower triangle. In addition, the height of the window can be reduced because the top of the triangle (near the VP) is too far away. Hence, the height of the triangle is reduced such that a distance of about 5 meters from the user is covered. The resulting trapezoid with parallel top and bottom lines is called the obstacle window (OW). For obstacles in the immediate vicinity beyond the reach of the white cane we have to consider distances between 2 and 5 m in front of the user, taking into account the height of the camera and perspective projection of the lens. However, the resolution at the bottom of the image is higher than that at the top or even at the VP. Therefore, we define a new obstacle detection window (ODW) and use interpolation to correct image resolution. Hence, the trapezoidal OW is converted to the rectangular ODW by maintaining the resolution at the top of OW but reducing it at the bottom. At this point we must stress that our system will not detect obstacles at a distance less than 2 m from the user, because of two reasons: (1) The user has already been alerted to a looming obstacle at a larger distance and advised to adapt path trajectory; (2) The user will always check a detected obstacle by using the white cane at short distance.

For obstacle detection in the ODW we apply three algorithms: (1) counting zero-crossings of image derivatives, (2) histograms of binary edges, and (3) Laws' texture energy masks. For further details please see [7]. If an obstacle is detected (a) in at least 3 consecutive frames, (b) by at least two of the three algorithms in each frame, and (c) with obstacle regions whose intersections are not empty, the user will be alerted and informed about the obstacle's position.

In addition, in order to avoid the obstacle, the user is instructed to turn a bit left or right. This is done by comparing the obstacle's region with the open spaces to the left and right in the path window. This is illustrated in Fig. 2, the rightmost image on the top row, where in this case the distances from the obstacle to the path borders are about equal and the system can randomly choose which side to go.

3. Lateral doors

While walking in corridors and on sidewalks, the user can be informed about typical objects like doors of rooms and houses. This gives the user an impression of where he or she is. In general, detection of doors is not quite trivial because there are different types with different frames, also with different geometries if viewed non-orthogonally.

Here we focus on doors in corridors. Assuming that the user walks along the corridor, guided by the vision module in centering towards the corridor's end, most doors are viewed laterally when they are approached. This complicates their geometry, although three things can be assumed: doors have vertical frames, their height is larger than their width, and they connect to the floor. The latter means that they can be found close to the corridor (path) borders as explained in the previous section.

We assume that the starting point is in a corridor, on one of the two sides, and that the corridor's two borders on the floor have been detected. As the user moves along the corridor, a door enters the camera's field of view almost in the center but at a large distance such that it is rather small. Depending on its distance, there may not be enough information to confirm its geometry. So we call it a candidate which will be tracked while approaching it. As the distance decreases, more information becomes available and detailed descriptors can be applied to confirm if the candidate is actually a door or for example the entrance of another, perpendicular corridor. Tracking is important because in some frames a door may be occluded, for example by another person.

After the Canny edge detector, the result of which is already available because of path detection, the first step is to extract long vertical edge segments that run upwards starting at the floor, as these are likely parts of door frames.

We look for doors on both the left and right sides if the horizontal distances between the position of the VP and the leftmost and rightmost point of the PW are bigger than 50 pixels. If one distance is smaller, the corresponding side will not be analyzed because it will not contain enough wall information. In addition, vertical edges are only checked in the left and right regions from the image borders to 3/4 of the distance to the VP, excluding also the path window; these are the light gray regions in Fig. 1 (top-left). As the image frames are distorted a bit due to the lens of the camera, which is more prominent close to the left and right image borders, and camera alignment may not be perfectly vertical due to the walking user, valid edge segments usually found at angles varying between 5 degrees from vertical.

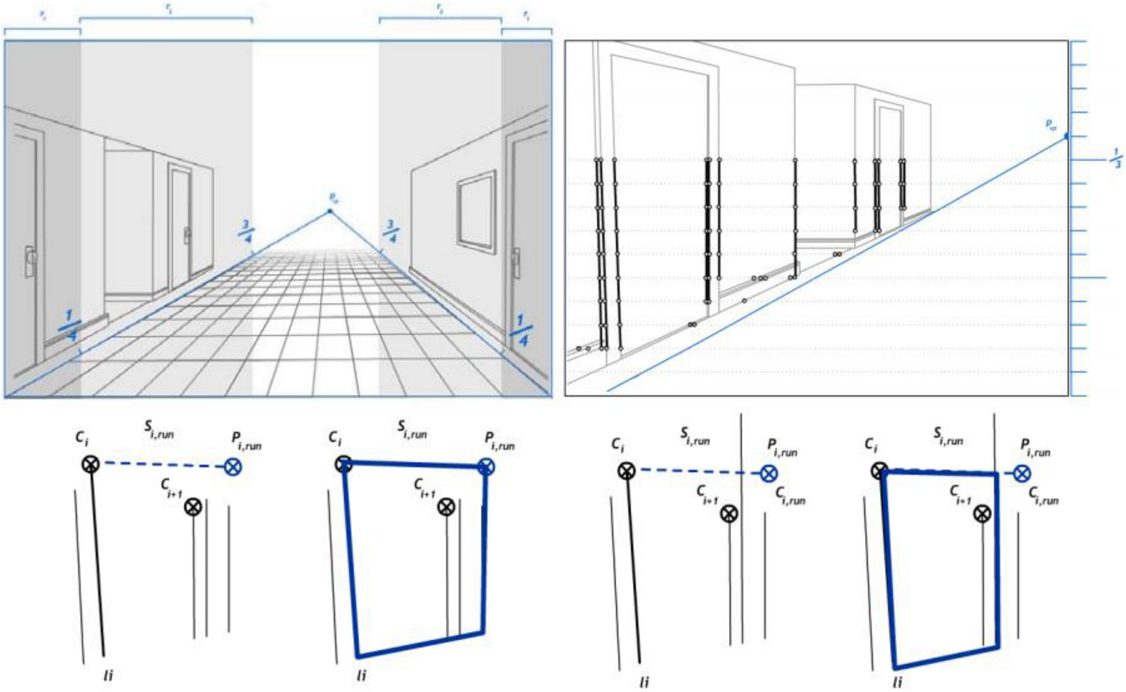

Fig. 1. Top: at the left the regions in which doors are detected, and at the right the door-detection algorithm. Bottom: detection of quadrilaterals; see text.

For detecting the edge segments, the positions of edge pixels are stored as nodes in a graph, but all edges located inside the Path Window are ignored. The graph is constructed by checking all edge pixels on image lines, from left to right and top to bottom. But instead of analyzing all image lines, we first only select every tenth line and start at 2/3 of the image height. Considering a horizontal interval of ±5 pixels to the left and to the right of each edge pixel, we check the next 10th line for edge pixels in these intervals. If an edge pixel is found, a new node is created and linked to the edge pixel on the above 10th line. Figure 1 (top-right) illustrates the lines checked and the created nodes where vertical edges are present. Edge segments with less than three connected nodes can be discarded because they are not long enough.

The created graph contains linked edge segments, but it only covers every 10th line of the image below 2/3 of the image height. The next step is to verify vertical edges between the sampled lines. The top and bottom nodes of each edge segment define a straight line, and edges in the binary edge map are checked along this line. Starting in the middle of the line and going up and down until the end points, continuity of edges in the edge map is checked by applying a distance tolerance of one pixel on each side. Small gaps are filled by interpolating the nearest, confirmed edge positions. Vertical edges below the path borders are not checked, but those above 2/3 of the image height are in order to obtain the most complete edge information. The result is a list of confirmed and significant vertical edges. Since short vertical edges are not likely part of door frames, they are discarded. Taking into account perspective projection, the vertical distance between the path border on the floor and the corresponding (left or right) border of the ceiling can be easily computed because they all pass through the VP.

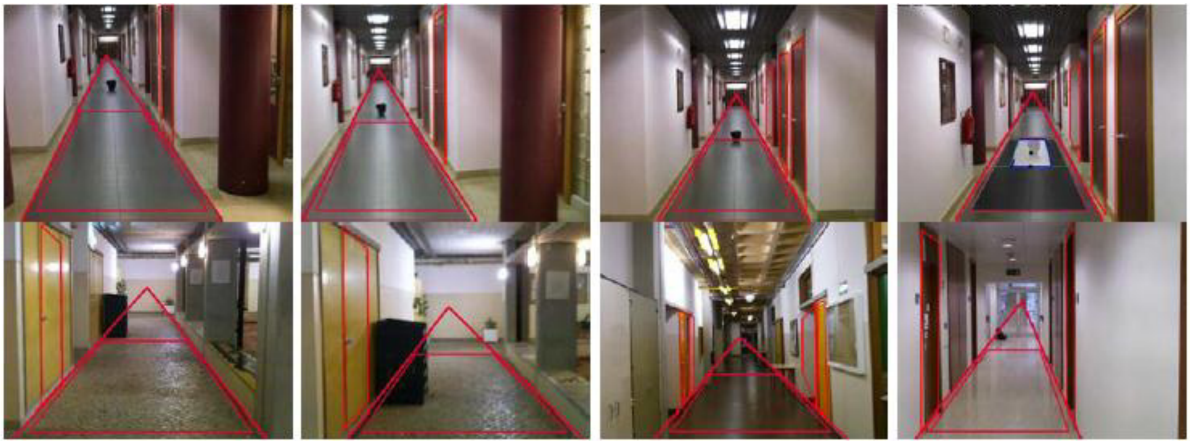

Fig. 2. Top: path, obstacle and doors detected along a corridor. Bottom: more examples in other corridors.

This distance is a linear function of the distance to the camera. All vertical edges with a length shorter than 2/3 of their corresponding distance are ignored.

Having a list of long vertical edges, quadrilateral geometry must be checked for confirming door frames. To this end we first check the top points of the vertical edges for corners. We apply the Harris corner detector [10] within a window of size $CR_{size} = 5{\times}7$ pixels. The Harris corner detector is based on the fact that a corner causes strong partial derivatives around a point of interest. If the two eigenvalues in the so-called Harris matrix H are approximately similar and above a threshold which depends on the size of the window, there is likely a corner present. The response of the detector is based on the approximation $CR = \lambda_x \lambda_y \approx det(H) - k * trace(H)$ with k=0.8. Let the points $P_{i,top}$ and $P_{i,bottom}$ be the top and bottom of the edge I and the local maxima of the corner detector at $P_{c,i,top}$. If the response at $P_{c,i,top}$ is higher than a threshold (10^{-6}) a corner is assumed and $P_{i,top}$ location is corrected to $P_{c,i,top}$. After all top points of vertical edges are checked for corners, an edge-tracking procedure is started at each corner in the direction of the vanishing point, with a vertical tolerance of 5 pixels. This procedure computes the mean pixel value $M_{c,v,p}$ at the start point $P_{c,i,top}$. It stops when the mean at a current point (x,y) deviates too much, i.e., $|M_{c,v,p} - M_{x,y}| > T_M$ with $T_M = 40$, or when there is a run of 3 consecutive missing edge pixels. If stopped at a point $P_{i,run}$, it forms, starting at $P_{c,i,top}$, an edge segment $S_{i,run}$.

A quadrilateral is formed by searching for the top of the vertical edge which is nearest to $P_{i,run}$; see Fig. 1 (bottom-left). If the nearest top position is above $P_{i,run}$, see Fig.1 (bottom-right), the edge is trimmed to the intersection point. The same process is repeated for all i.

A quadrilateral found in one frame may not be detected in a next frame. This can happen if there is no top “horizontal” edge due to excessive camera tilt, or because of poor or excessive light sources which can hide real edges or create shadow edges. For these reasons quadrilaterals, once detected, are tracked in subsequent frames, and their characteristics can be determined with higher accuracy when they are approached. The quadrilaterals’ bottom points close to the floor, therefore close to the path border, are good indicators for tracking. In principle, all bottom points are collinear, for example four points of two doors, and their distance ratios in two consecutive frames are preserved in perspective projection. Hence, tracking over frames could work if there are at least four matching edges in two consecutive frames. However, this is unlikely to happen, mainly because of perspective occlusion of the closest inner edge between door surface and door frame by the frame itself, even if the surface and frame have different colors.

Therefore the following approach was adopted. If a quadrilateral is found in region r_1, see Fig.1 (top-left), the search for the quadrilateral in the next frame will begin in the same region of the quadrilateral's bottom points, considering a displacement of three pixels in both directions along the path border. If a corner of the quadrilateral enters region r_2, a template patch of the bottom point is saved for further localization. The same interval plus displacement is considered, but each bottom point of vertical edges is tested against the patch.

Figure 2 shows examples of corridors with detected path borders and doors in red. The top row shows door detection and tracking while approaching an obstacle in the center of the corridor. The bottom row shows more examples of door detection. In the right two frames we can see that some doors have not been detected, because they are too close or too far. Nevertheless, these are detected in other frames due to the tracking, such that the user can be informed about their existence.

4. Sound and Speech interfaces

Apart from the white cane, blind persons have learned to rely on the sounds surrounding them; e.g. [11]. In fact, the brains of most blind have learned to devote a big part of the visual cortex to audition, which is called sensory substitution. For these reasons we cannot use normal headphones, as these block surround sounds. Instead we use either the loudspeaker of the computer or bone-conducting headphones [12] which are normally used by swimmers in pools. If the latter option is too expensive, a normal headphone which does not block all surround sounds can be used.

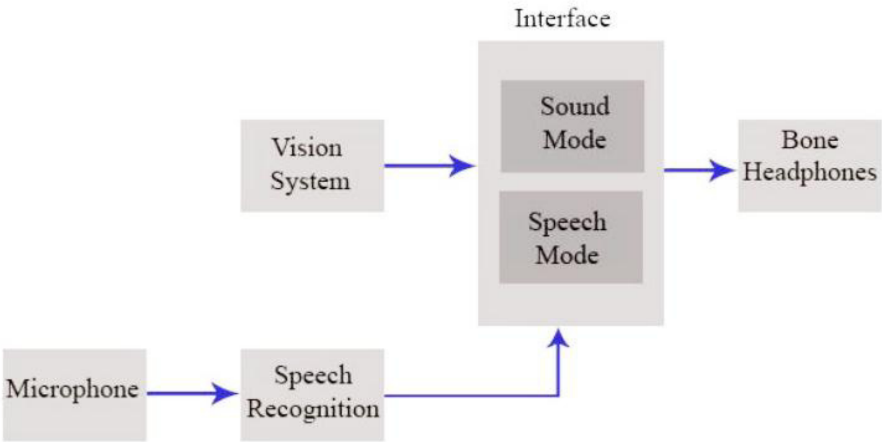

Fig.3. Structure of the user interface.

In either case the synthesized sounds must be quite different from normal sounds and their volume must be adjustable.

In addition, different persons have different preferences, so we offer a large set of sounds to choose from, and these can be selected for different purposes. This process is part of the installation and should not be done while using the system.

Figure 3 shows the structure of the audio interface. Both sound and speech can be synthesized. Input consists of information from the vision module and a microphone for speech recognition. While using the system, the user can select the sound mode or the speech mode. Speech recognition can be activated by pressing a special key, in order to interpret vocal commands of the user. The information of the vision module consists of very few coded messages: valid path found or not found, path centering is OK or should be corrected to the left or right, obstacle detected at *N* meters, avoid obstacle by going a bit to the left or right, plus a few messages devoted to special events like door detected on the left/right side. The sound and speech synthesizers only translate these messages into audible signals.

Likewise, speech recognition concerns only very few commands, for starting and stopping sound or speech synthesis, for selecting the sound or speech output mode, and for controlling the volume. All commands from the user and messages for the user have been reduced to an absolute minimum in order to simplify the system's usage and to avoid any confusion.

The module for speech recognition employs the Hidden Markov Model Toolkit (HTK) [13]. It supports two modes of speech recognition, speaker-dependent and speaker-independent, and it can be used for continuous speech or for individual words. Our interface is devoted to individual words and is by default speaker-independent. The advantage of speaker-independent recognition is that the user does not need to train it. The set of voice files contains 640 wave files of 8 words repeated 10 times by 8 different persons. The word set contains: stop, go[*], mute, louder, lower, repeat, sound and speech. In the tests we conducted, the recognition rate in the case of speakers not included in the training set was very good. However, if a user experiences many problems, he or she is advised to switch to speaker-dependent mode and to train the system using the own voice. This has the additional advantage that a different language can be used.

As mentioned above, the user can select sound mode and speech mode. Sounds are synthesized using pulse-code modulation (PCM) in a mono channel, with 16-bit resolution and a sampling rate of 44.1 kHz. We used the Open Audio Language (OpenAL) for generating sounds in the Waveform Audio File (WAVE) format. The sounds have a fundamental frequency of 880 Hz, which corresponds to the tone A which is often used for tuning musical instruments (a normal tuning-fork of 440 Hz is one octave lower). However, all sounds can be modulated in amplitude (louder if an obstacle becomes closer) and in frequency (a higher frequency indicating left and a lower one right). All sounds have duration of 1s, they can be modulated by an envelope function with ADSR parameters, and they can be repeated, or some can be played continuously. For example, path centering can employ a continuous sound, and a looming obstacle can be signaled by a repetitive short sound. The system uses a few default sounds, but the user can select a few sounds from a large set, for example waves which have different spectra: sinusoidal for path centering (sinusoidal sounds are rather neutral), triangular for obstacles (these are more conspicuous because of higher harmonics), and rectangular if no valid path can be found (these sound very harsh: attention!).

Figure 4 illustrates several situations, with the corresponding default sound waves at the bottom. The left three images on the top row illustrate path centering. In the second image, path centering is perfect: a default sinusoidal tone at 880 Hz. In the first and third images, the path center is detected to the left and to the right, respectively, as indicated by the higher and lower frequency. The rightmost image on the top row and the left three images on the second row show a few frames in which an obstacle is approached: the sound wave changes from sinusoidal to triangular, and both amplitude and frequency increase. In the last image no valid path can be detected and a harsh squarewave sound is played. In speech mode, the same information as explained above is transmitted by synthesized speech messages.

We use Google's text-to-speech synthesizer to generate these messages in several languages: Portuguese, English, French, Spanish and German. Examples of messages are: "path ok" when a valid path in front of the user has been detected; "turn left/right a bit" when the user deviates a little bit from the path's center line; "path lost" when no valid path can be detected (this is repeated until a valid path is encountered); "turn left/right to go back to the valid path" when the user leaves the valid path but it can still be detected; and "obstacle at *x* meters"

[*] Go is used instead of start because is difficult to discriminate from stop.

once an obstacle has been detected together with "turn left/right to avoid it". The latter is repeated every few seconds or if the distance to the obstacle has changed.

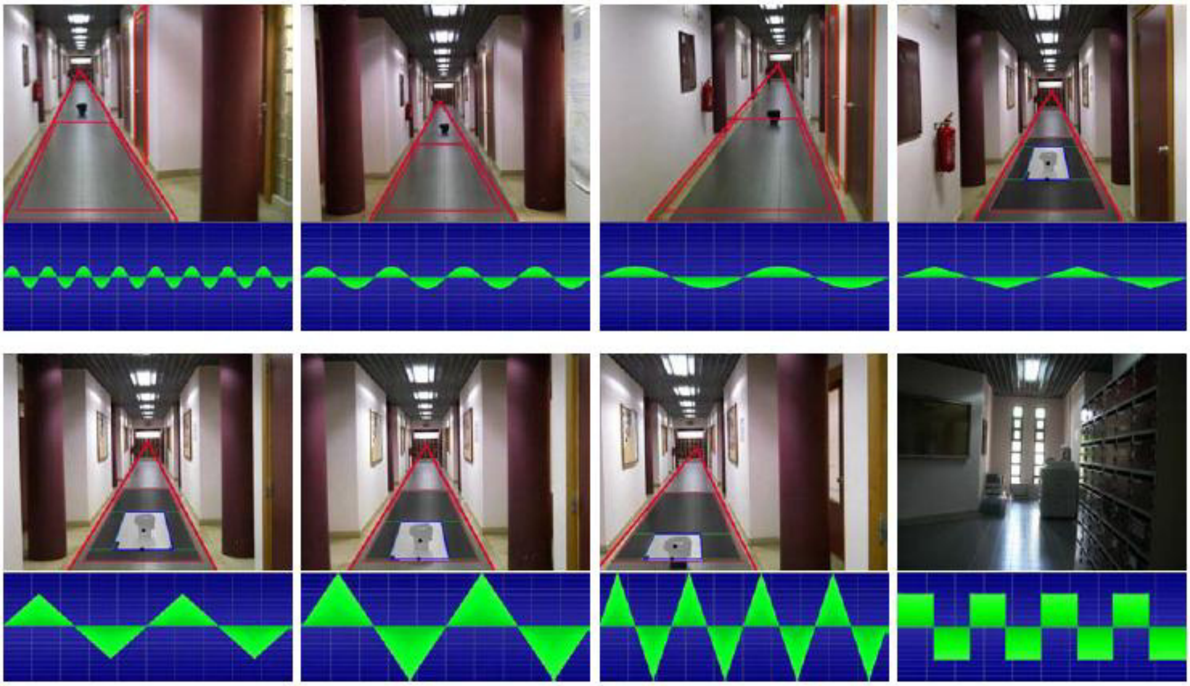

Fig. 4. Examples of path and obstacle detection with sound synthesis. From left to right and top to bottom: the first three images show path centering with a frequency-modulated sinusoidal sound. The next four images show a sequence in which an obstacle is approached, as indicated by a triangular sound increasing in amplitude and frequency. In the last image, no valid path can be detected and a harsh sound is played.

5. Conclusions

We presented a vision system that detects lateral doors in corridors. Although in some frames a door may be missed, the tracking algorithm assures that the door will be detected in subsequent frames. This improves the reliability of the system and the user can be informed about nearby doors while walking in a corridor. Ongoing work concerns detection of the door type, as doors can contain windows or not. Also, normal doors have a knob at the left or right side, approximately at waist height, whereas fire doors may have a horizontal bar. Additional information consists of signs on or next to doors, reading WC, EXIT, STAIRS and ELEVATOR, also including a room /door number or name.

Basically, the goal is to detect all useful information for the user, such that he or she can become more familiar with a building or find a specific destination or room [5]. Although similar to door detection in corridors, detection of doors of houses and entrances of buildings, for example when walking on a sidewalk, is much more complicated because of the larger variability.

We also presented the sound interface that assists the user in centering on paths like sidewalks and corridors, alerting to looming obstacles for avoiding them. The user can choose between sound and speech mode, with a large collection of different sounds of which a few must be selected. This allows the user to customize the interface according to his personnel preferences. Speech and sound synthesis is complemented by speech recognition, but, as for speech and sound synthesis, limited to very few words. The interface is optional because the user can prefer not to use sounds, but a vibrating bracelet or cane handle instead [6].

The prototype system is completely based on off-the-shelf components, i.e., camera, microphone, headphones or perhaps bone-conducting phones, a cheap notebook computer being the most expensive item. A manual is being prepared such that the system can be assembled and installed by any technician. Nevertheless, the same system is also being adapted to a SmartPhone which can be worn with a strap around the neck. Although a suitable SmartPhone may still be too expensive, it is expected that in two or three years the price will be much lower.

An initial prototype of the system has been already tested by a blind person, and many more tests are planned. These tests will be conducted in collaboration with the Portuguese association for the blind and amblyopes (ACAPO), which participates in the Blavigator project.

Acknowledgements

This work was partially supported by the Portuguese Foundation for Science and Technology (FCT) project PEst-OE/EEI/LA0009/2011, FCT project Blavigator RIPD/ADA/109690/2009 and PhD grant SFRH/BD/82310/2011.

References

[1] World Health Organization, Visual impairment and blindness, Fact Sheet n._282, October 2011; URL: http://www.who.int/mediacentre/factsheets/fs282/

[2] Singh, V., Paul, R., Mehra, D., Gupta, A., Sharma, V., Jain, S., Agarwal, C., Garg, A., Gujral, S., Balakrishnan, M., Paul, K., Rao, P., Manocha, D., "Smart" cane for the visually impaired: Design and controlled field testing of an a_ordable obstacle detection system, in: In Proc. 12th Int. Conf. on Mobility and Transport for Elderly and Disabled Persons(TRANSED); 2010, Hong Kong.

[3] Schmitz, B., Becker, S., Blessing, A., Matthias, G., Acquisition and presentation of diverse spatial context data for blind navigation, In Proc. IEEE 12th Int. Conf. on Mobile Data Management; 1, 2011, p. 276 – 284.

[4] Costa, P., Fernandes, H., Vasconcelos, V., Coelho, P., Barroso, J., Hadjileontiadis, L., Landmarks detection to assist the navigation of visually impaired people, In Proc. 14th Int. Conf. on Human-computer interaction: towards mobile and intelligent interaction environments, 3, 2011, p. 293 – 300.

[5] Tian, Y., Yang, X., Arditi, A., Computer vision-based door detection for accessibility of unfamiliar environments to blind persons, 12th Int. Conf. on Computers Helping People with Special Needs LNCS 6180; 2010, p, 263–270.

[6] du Buf, J., Barroso, J., Rodrigues, J., Paredes, H., Farrajota, M., Fernandes, H., Jose, J., Teixeira, V., Saleiro, M., The SmartVision navigation prototype for blind users, JDCTA: International Journal of Digital Content Technology and its Applications 5 (5); 2011, p. 351 – 361.

[7] José, J., Rodrigues, J., du Buf, J., Visual navigation for the blind: path and obstacle detection, In Proc. Int. Conf. on Pattern Recognition Applications and Methods (ICPRAM 2012) 2; 2010, p. 515–519.

[8] Canny, J., A computational approach to edge detection, IEEE Trans. Pattern Anal. Mach. Intell. 8, 6; 1986, p. 679–698.

[9] Duda, R., Hart, P., Use of the Hough transform to detect lines and curves in pictures, Comm. ACM 15; 1972, p. 11–15.

[10] Harris, C., Stephens, M., A combined corner and edge detector, In Proc. 4th Alvey Vision Conference; 1988, p. 147 – 151.

[11] Wan, C., Wood, A., Reutens, D., Wilson, S., Early but not late-blindness leads to enhanced auditory perception, Neuropsychologia 48 1; 2010, p. 344 – 348.

[12] Audio Bone, Audio Bone 1.0 - The Ear Bud Reinvented; URL: http://www.audioboneheadphones.com/

[13] Young, S. J., Evermann, G., Gales, M. J. F., Hain, T., Kershaw, D., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., Woodland, P. C., The HTK Book, version 3.4, Cambridge University Engineering Department; 2006, Cambridge, UK.